universidade anhembi morumbi
Mundialmente criativa e inovadora
Laureate International Universities®

Programação para Jogos 2D

Prof. Alexandre Nascimento Tomoyose

1º semestre 2017

universidade anhembi morumbi
Mundialmente criativa e inovadora
Laureate International Universities®

Orientação a Objetos

Prof. Alexandre Nascimento Tomoyose

1º semestre 2017

Orientação a Objetos

- **Encapsulamento**
  - Conceito de restringir o acesso dos atributos e métodos de uma classe ao máximo para proteger seu estado interno
  - Aumenta a confiabilidade do código, pois bloqueia ações duvidosas já na compilação

Orientação a Objetos

- **Encapsulamento**
  - As próprias classes já são uma forma de encapsulamento
  - Pode ser refinado utilizando palavras-chave que controlam o nível de acesso:
    - private (apenas a própria classe)
    - protected (própria classe e subclasses)
    - public (acesso livre)

Orientação a Objetos

- **Classe Abstrata**
  - Classe que não pode ser instanciada
  - É possível definir atributos e métodos em uma classe abstrata, mas para usá-los é necessário realizar herança
  - No código, utilizamos a palavra-chave **abstract**

Orientação a Objetos

- **Classe Abstrata**
  - **Exemplo conceitual:**
    - **Classe abstrata Felino**
    - **Subclasse Gato que herda de Felino**
    - **Subclasse Tigre que herda de Felino**
  - **Neste caso, não é possível instanciar um Felino por ser uma classe abstrata, mas é permitido instanciar Gato ou Tigre, subclasses de Felino**

Orientação a Objetos

- **Exercício 1**
  - **Criar classe Forma**
    - **abstrata**
    - **atributos: x, y, largura, altura, cor**
    - **método construtor gera valores aleatórios para os atributos**

Orientação a Objetos

- **Exercício 1b**
  - Criar classe Retangulo
    - herda de Forma
    - método construtor recebe como parâmetros os valores de largura e altura
    - método desenha() faz um retângulo com os atributos
  - Função setup()
    - criar e desenhar uma instância de Retangulo

Orientação a Objetos

- **Super / Base**
  - Palavra-chave que referencia a classe pai (superclasse)
  - Na linguagem Java é utilizado o termo **super** enquanto no C# a palavra-chave é **base**

Orientação a Objetos

- **Super / Base**
  - **Permite acessar métodos da classe pai a partir da classe filha**
  - **No construtor é utilizado para definir os parâmetros a serem usados para construir a classe pai**

Orientação a Objetos

- **Exercício 1c**
  - Criar classe Quadrado
    - herda de Retangulo
    - método construtor recebe como parâmetro a medida do lado
  - Função setup()
    - criar e desenhar uma instância de Quadrado

Orientação a Objetos

- **Atributos / Métodos estáticos**
  - Define quais atributos e métodos devem ser considerados comuns da classe, não somente de um objeto instanciado
  - Propriedades estáticas podem ser acessadas diretamente da classe, não é necessário instanciar

Orientação a Objetos

- **Atributos / Métodos estáticos**
  - **No caso de atributos estáticos todas as instâncias compartilham a mesma variável, ou seja, possui um único valor para todas elas**
  - **Para definir uma propriedade como estática utilizamos a palavra-chave static**

Orientação a Objetos

- Atributos / Métodos estáticos
  - Muito usado para criação de classes utilitárias
  - Exemplo:
    - Classe Matematica
      - Matematica.random()
      - Matematica.pi

Orientação a Objetos

- **Exercício 2**
  - **Criar classe Matematica**
    - **atributo estático pi com valor 3.1416**
    - **método estático fatorial() que recebe como parâmetro um número inteiro e retorna o resultado da operação fatorial**
  - **Função setup()**
    - **escrever no console o valor de pi**
    - **escrever no console o resultado do fatorial de 5, 10 e 50**